ÉCOLE NATIONALE SUPÉRIEURE DE TECHNIQUES AVANCÉES BRETAGNE
FORMATION D'INGÉNIEUR SOUS STATUT ÉLÈVE

BERNARDO HUMMES FLORES

Robot Localization using Interval Analysis on Tiled Floors

Report presented in order to obtain a partial grade in the 2nd year internship evaluation in Autonomous Robotics.

Prof. Dr. Luc JAULIN
Advisor

Brest
September 2021

ACKNOWLEDGEMENT

I would like to express my gratitude to professor Luc Jaulin, that helped making possible not only my internship, but also my coming to ENSTA Bretagne despite COVID restrictions. Also, for the orientation and teachings that permitted to conduct this project.

I want to thank as well professor Mariana Kolberg, that helped me with guidance and in realizing this exchange.

In addition, I'd like to extend my thanks to the professors and friends whose discussions lead me to overcome problems of the moment.

Résumé

Le bon fonctionnement des robots mobiles sans supervision nécessite une estimation correcte de sa position, ce qui est rendu difficile par des capteurs qui ont des bruits et du matériel restreint. Les méthodes basées sur l'analyse par intervalles offrent un moyen de résoudre ce problème avec des garanties intéressantes, comme l'impossibilité de présenter de faux résultats, qui sont des alternatives viables lorsqu'elles sont suffisamment précises. En même temps, ils permettent l'exploration de relations fortement non linéaires et difficiles à représenter d'une manière élégante et efficace. Ce rapport présentera une méthode de localisation garantie développée pour les environnements avec des sols carrelés. Elle a conduit à la validation de la théorie, à l'implémentation d'une localisation opérationnelle dans des scénarios limités, à des informations sur l'efficacité des méthodes basées sur les intervalles et à une plateforme de test avec et sans ROS, ainsi qu'à une simulation du robot dans l'environnement carrelé.

Abstract

The correct operation of mobile robots without supervision requires a correct estimation of its position, which is made difficult with noisy sensors and restricted hardware. Methods based on interval analysis offer a way of solving this problem with interesting guarantees, such as the impossibility to present false results, that are viable alternatives when sufficiently precise. At the same time, they allow for the exploration of heavily non-linear and hard to represent relations in an elegant and efficient way. This report will present a guaranteed localization method developed for environments with tiled floors. It has lead to a validation of the theory behind it, an operating localization implementation within limited scenarios, insights on the efficiency of interval based methods, and a testing platform with and without ROS, alongside a simulation of the robot in the tiled environment.

CONTENTS

1 CONTEXT

The developed internship has been done at Lab-STICC, a laboratory based at Brittany, in France, that deals with a broad spectrum of topics on information and communication technologies in multiple sectors, such as defense, security, maritime environment and robotics. My activities were a part of the robotics branch, ROBEX - ROBotics for EXploation, a research team inside ENSTA Bretagne that focuses on maritime applications of autonomous robots.

My participation in ROBEX has been contributing to research on robot localization, developing a project proposed by professor Luc Jaulin, a localization method for robots using a single camera facing downwards in a place with tiled floors. This project is interesting both because of its theoretical basis and the meaning of its application in a real world problem. Projects in ROBEX often deal with interval analysis, topic that will be better explained in section 3.1, but that represents an alternative approach that needs justifying for choosing. This project enters by displaying a real world example of simplicity and efficiency when compared to more traditional approaches, offering guarantees about the final solution that do not exist in others, such as probabilistic methods that are susceptible to diverge.

In those 4 months of activities, I have deepened my theoretical knowledge on interval analysis and image processing in robotics, necessary for the understanding and expanding on the researched topic. Requirements of the project also made me learn more about simulating robotic systems and creating testing platforms for validating ideas, as well as how to balance the levels of precision and efficiency. By dealing with several setbacks on the progress, either with failing ideas or unexpected findings, I was familiarized to the difficulties of the research process and had to improve my adaptability while exercising constant critical thinking. Finally, the importance of a clear communication was made evident, and a development on the capacity of doing so was required for effectively discussing with my project advisor the problems and successes of each step.

2 INTRODUCTION

The localization of mobile robots is an active research field, fundamental for safely and correctly performing tasks without human supervision, activities that are usually dependent on a precise estimation of the robot's state. Many sensors are capable of providing data for such tasks, but recent progress has made cameras specially accessible, and a great option when considering the amount of information made available from them [3]. They are also interesting for capturing patterns presented in the environment that would be difficult otherwise to perceive, being that the case scenario of the proposed method, where tiles with repetitive shapes compose the floor of the entire area of operation, thus making available guarantees on what can a camera sensor may capture when moving with this view.

Localization methods based on interval analysis have existed for a while, but they do encounter some resistance in acceptation from the community as a viable option. Part of this is due to the pragmatically different approach that it takes, representing uncertainty as an area with infinite distributions of probability and its fundamental guaranteed results that may be seen as too costly to compute, which is in constant improvement [4]. With this in mind, this method has came to existence as an example of how a tangible and understandable real world problem may have an efficient resolution in this approach.

Algorithms for solving the localization problem diverge, initially, in those that make available the initial position and those that do not [5], also known as the kidnapped robot problem. They also diverge in the types of sensors used, such as dead reckoning [6] with specially precise hardware, lasers [7, 8], sonars [9, 10], cameras [3, 11, 12, 13], among others. On the matter of techniques used, the most common approach is based on probabilistic methods [5], but alternative takes exist, such as the ones based interval analysis and hybrids [14, 15] that try to take advantage of both worlds. This work will present an interval localization method with known initial position, based mainly on a camera sensor, but also using proprioceptive measurements of speed and heading.

This report will continue with a necessary theoretical background needed for understanding in chapter 3. Chapter 4 will explain the image processing steps used in the method, and the explanation of the localization method will be done in chapter 5. The implementation of the aforementioned techniques will be specified in chapter 6, explaining the architecture used and the testing environments. Chapter 7 will present the results achieved in this work. Finally, chapter 8 will present some concluding words and what is thought for the future work.

The code for the method and simulation can be found in the following repository: https://github.com/birromer/tiles-localization.

3 THEORETICAL BACKGROUND

3.1 Interval analysis

The usage of intervals to represent uncertainty provides the guarantee that results will not diverge [16], a problem often found in probabilistic methods. This property, alongside their capacity of dealing with non-linear constraints and the availability of contractors to be used within this method lead it to be chosen for this project.

An interval $[x]$ may represent a subset of $\mathbb{R}$, with its limits defined by upper and lower bounds. This notion can be extended for multiple dimensions, allowing for the representation of multidimensional data with subsets of $\mathbb{R}^n$, such as

$$[\mathbf{x}] = [x_1] \times [x_2] \times \cdots \times [x_n] \tag{3.1}$$

with

$$[x] = [x^-, x^+] = \{x \in \mathbb{R} \,|\, x^- \leq x \leq x^+\} \tag{3.2}$$

From this basis, it is possible to extend the application of basic arithmetic operations on intervals. Operations on numeric values $[x]$ and $[y]$ with binary operations can be expressed by

$$[x] \diamond [y] = \{x \diamond y \in \mathbb{R} \,|\, x \in [x], y \in [y]\} \tag{3.3}$$

These ideas can be extrapolated to functions, which then allow for the construction of contractor operators, capable of removing unfeasible solutions that don't respect a series of constraints. An example is the add contractor, $\mathscr{C}_+$, which removes the solutions from the sets $[a]$, $[b]$, $[c]$ that don't respect the constraint $a + b = c$, represented as $a + b - c = 0$, which are

$$\begin{cases} A + B = C \\ C - B = A \\ C - A = B \end{cases} \tag{3.4}$$

3.2 Tiles

3.2.1 Parametrization

Tiled floors are repetitive structures, with ambiguity in its definition along the X and Y axis, as well as rotation-wise. They may be represented by the sets satisfying

$$T_0(a_1, a_2) = \sin \pi a_1 \cdot \sin \pi a_2 = 0 \tag{3.5}$$

They may be described by the set of parameters $\mathbf{y}$ extracted from a view, and they correspond to the horizontal displacement, the vertical displacement and the rotation applied to them, respectively. If those parameters are applied in reverse order on the image, the original orientation and origin centered tiles would be the result. Because of the multiple ambiguities on their view, the domains are defined as follows, $y_1, y_2 \in [-1/2, 1/2]$ and $y_3 \in [-\pi/4, \pi/4]$, where y_1 and y_2 repeat every half length, and y_3 repeats every $\pi/4$ turn. With those, a tile with parameters $\mathbf{y}$ may be defined as

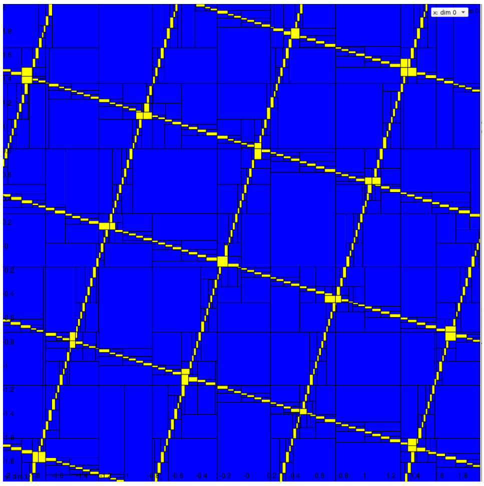
$$\mathbb{T}_y = \{a | T_y(a) = 0\} \tag{3.6}$$

where

$$\mathbb{T}_y(a) = \mathbb{T}_0 \begin{pmatrix} y_1 + a_1 \cos y_3 - a_2 \sin y_3 \\ y_2 + a_1 \sin y_3 + a_2 \cos y_3 \end{pmatrix} \tag{3.7}$$

The following figure figure illustrates the view associated with tiles parametrized by $\mathbf{y} = (0.1, 0.2, 0.3)$:

Figure 3.1: Frame with associated parameters $y = (0.1, 0.2, 0.3)$.



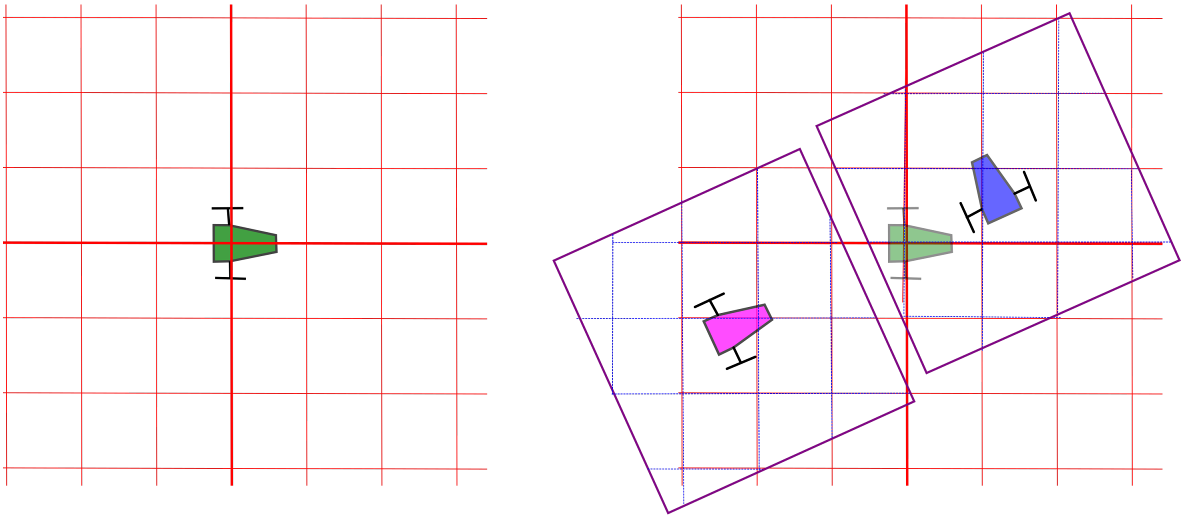
3.2.2 Equivalency

Multiple views of the tiles may generate the same parameters, and an equivalence relation can be established in order to link them:

$$\mathbf{y} \sim \mathbf{z} \Leftrightarrow \mathbb{T}_{\mathbf{y}} = \mathbb{T}_{\mathbf{z}} \tag{3.8}$$

This equivalence can be seen in the following image.

Figure 3.2: The blue and pink robots see the same frame, image credit of professor Luc Jaulin.



Because of that, an equivalency between two sets of parameters $\mathbf{z}$ and $\mathbf{y}$ can be expressed by the following theorem:

Theorem 1 *We have*

$$y \sim z \Leftrightarrow \begin{cases} y_1 - z_1 \in \mathbb{N} \\ y_2 - z_2 \in \mathbb{N} \\ \frac{y_3 - z_3}{\pi} \in \mathbb{N} \end{cases} or \begin{cases} y_1 - z_2 \in \mathbb{N} \\ y_2 - z_1 \in \mathbb{N} \\ \frac{1}{2} + \frac{y_3 - z_3}{\pi} \in \mathbb{N} \end{cases} \tag{3.9}$$

Equivalently, we have

$$y \sim z \Leftrightarrow \begin{cases} \sin \pi (y_1 - z_1) = 0 \\ \sin \pi (y_2 - z_2) = 0 \\ \sin (y_3 - z_3) = 0 \end{cases} or \begin{cases} \sin \pi (y_1 - z_2) = 0 \\ \sin \pi (y_2 - z_1) = 0 \\ \cos (y_3 - z_3) = 0 \end{cases} \tag{3.10}$$

Proof.

1. $y_i - z_i \in \mathbb{N} \Leftrightarrow \sin(\pi(y_i - z_i)) = 0$

$$\begin{aligned}
\sin\theta = 0 &\Leftrightarrow \theta = k\pi,\ k \in \mathbb{N} \\
y_3 - z_3 \in \mathbb{N} &\Rightarrow \sin((y_3 - z_3)\pi) = 0 \\
\sin\pi(y_3 - z_3) = 0 &\Rightarrow \exists k,\ k\pi = (y_3 - z_3)\pi \\
&\Rightarrow k = y_3 - z_3 \\
&\Rightarrow y_3 - z_3 \in \mathbb{N}
\end{aligned}$$

2. $\frac{y_3-z_3}{\pi} \in \mathbb{N} \Leftrightarrow \sin(y_3 - z_3) = 0$

$$\begin{aligned}
\sin\theta = 0 &\Leftrightarrow \theta = k\pi,\ k \in \mathbb{N} \\
\frac{y_3 - z_3}{\pi} \in \mathbb{N} &\Leftrightarrow \sin(\frac{y3_{-}z_3}{\cancel{\pi}}\cancel{\pi}) = 0 \\
&\Leftrightarrow \sin(y3_{-}z_3) = 0
\end{aligned}$$

3. $\frac{y_3-z_3}{\pi} + \frac{1}{2} \in \mathbb{N} \Leftrightarrow \cos(y_3 - z_3) = 0$

$$\begin{aligned}
\cos\theta = 0 &\Leftrightarrow \theta = \left(k + \frac{1}{2}\right)\pi,\ k \in \mathbb{N} \\
\frac{y_3 - z_3}{\pi} + \frac{1}{2} \in N &\Leftrightarrow cos\left(\left(\left(\frac{y_3 - z_3}{\pi} + \frac{1}{2}\right) + \frac{1}{2}\right)\pi\right) = 0 \\
&\Leftrightarrow \cos\left(\frac{y_3 - z_3}{\cancel{\pi}}\cancel{\pi} + \pi\right) = 0 \\
&\Leftrightarrow \cos(y_3 - z_3 + \pi) = 0 \\
&\Leftrightarrow \cos(y_3 - z_3 + \pi + \pi) = 0 \\
&\Leftrightarrow \cos(y_3 - z_3) = 0
\end{aligned}$$

The two options for equivalence exist because the robot does not have the information of which are the horizontal and which are the vertical lines, representing them a swap in the first two dimensions, as well as a 90° rotation of its orientation. Those equations take in consideration tiles with sizes of unitary length, but they can be normalized for working with and arbitrary size L, as follows:

$$y \sim\ z \Leftrightarrow \begin{cases} \sin\left(\pi \frac{y_1 - z_1}{L}\right) = 0 \\ \sin\left(\pi \frac{y_2 - z_2}{L}\right) = 0 \\ \sin\left(\frac{y_3 - z_3}{L}\right) = 0 \end{cases} \quad or \begin{cases} \sin\left(\pi \frac{y_1 - z_2}{L}\right) = 0 \\ \sin\left(\pi \frac{y_2 - z_1}{L}\right) = 0 \\ \cos\left(\frac{y_3 - z_3}{L}\right) = 0 \end{cases} \tag{3.11}$$

Finally, these relations will allow the estimated positions in the real world to be contracted with equivalent parameters extracted from the incoming camera view of the robot.

4 PARAMETER DETECTION

The first step in the proposed method is to extract the aforementioned parameters from the image in the view of the robot. This chapter will describe the treatment applied to the received frames, the cleaning steps and how the displacement of the tiles is computed from that.

4.1 Pre-processing

The objective of this pre-processing is to improve the detection of lines, as noise and other visual artifact may create false positives that need to be filtered out. All data used comes from simulated environments, allowing perspective and shadow complications to be ignored. However, there are still problems with the quality of the images used as floor, such as the cascaded rasterization of lines and overall resolution.

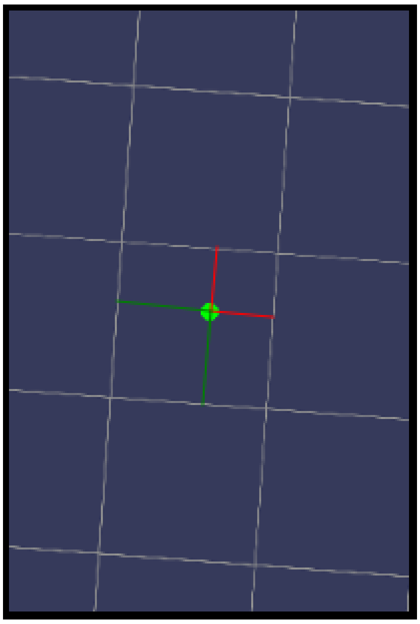
At the first moment, images are converted to grayscale for simplifying the treatment, this is done with a simple color space conversion from RGB to grayscale. The derivatives of each frame are computed with a Laplacian filter, enhancing the contours of the lines and improving the edge detection, which is done with the Canny filter [17], leading to a binary image. Because of the thickness of the lines, sometimes multiple edges appear around the area of a single line, as well as noisy patches are made evident. In order to unify the area of each line and clean those artifacts, the morphology operators of closing and dilatation are used.

The resulting image passes through the probabilistic Hough transform [18, 19], which detects the present lines and make available their starting and finishing points, respectively (x_1, y_1) and (x_2, y_2). Those points are used to compute the distance to the center of the image, (c_x, c_y), with the following equation, in order to obtain a distance d. That distance computation is represented in figure 4.1, the same process being applied to all lines.

$$d = \frac{|((x_2 - x_1) \cdot (y_1 - c_y) - (x_1 - c_x) \cdot (y_2 - y_1))|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}} \tag{4.1}$$

All pixel values are converted to meters according to evaluated constants and stored alongside other information in data structures for usage in the following steps. Later, the lines' orientations are used in order to filter possible outliers. This is done by

Figure 4.1: Distance computed from point to line, later used for extracting displacement. Red lines are the value d taken directly from equation 4.1, green lines are computed as $1-d$.



comparing each pair of points from the disambiguation function, which will be explained in section 4.2, with the median of all, $(\hat{x},\hat{y})$, with the limit of negligible ζ.

$$|\hat{x}-x_i|+|\hat{y}-y_i|<\zeta \tag{4.2}$$

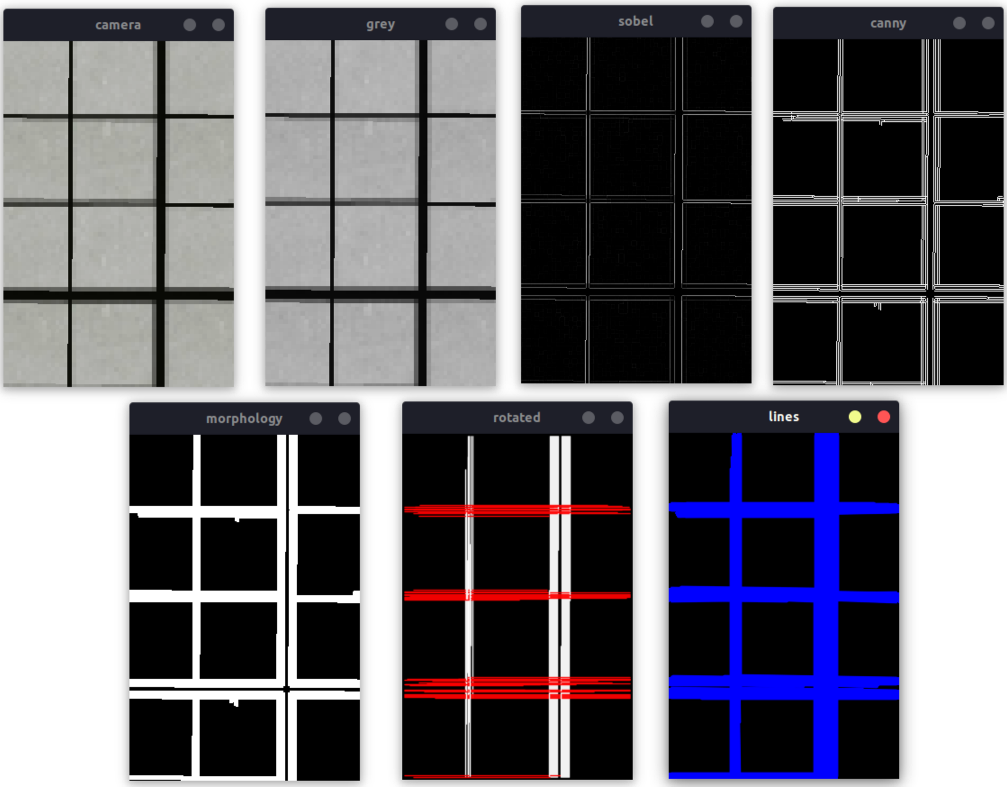
The result of the described steps can be seen in the figure 4.2.

4.2 Parameter extraction

The parameter extraction from a view relies mostly on the successful separation of the two sets of perpendicular lines and the estimation of their distances. The parameter vector $\mathbf{y}$ is extracted with the information on bags of vertical and horizontal lines. It is important to point that the notion of vertical and horizontal lines is limited to the separation of two bags and a local view of the floor, such information is not immediately known by the robot and must be estimated according to its movement.

Because of the fact that all angles have equivalents that are rotations of $\pi/2$ of one another, representing the same parameters, an equivalency between two angles can

Figure 4.2: Steps of pre-processing applied in the incoming image. First row: original image, grayscale, laplacian filter, canny filter. Second row: morphology operations, horizontal/vertical detection, final lines detection.



be described with

$$\begin{cases} cos\ 4\alpha_2 = cos\ 4\alpha_1 \\ sin\ 4\alpha_2 = sin\ 4\alpha_1 \end{cases} \tag{4.3}$$

which leads to the disambiguation function

$$\begin{pmatrix} x_i \\ y_i \end{pmatrix} = \begin{pmatrix} cos\ 4\alpha_i \\ sin\ 4\alpha_i \end{pmatrix} \tag{4.4}$$

From the point cloud composed of x_i and y_i we can then get the angles of the lines via

$$\hat{\alpha} = \frac{1}{4} \arctan 2(\hat{y}, \hat{x}) \tag{4.5}$$

Those lines are then separated in bags of vertical and horizontal lines, according to their angle, if the cosine or sinus are close to 0, respectively. Finally, each of the bags can generate a displacement of the tiles view by taking the median of the displacement between lines, dd, normalized by the size of the tiles, L, as seen bellow:

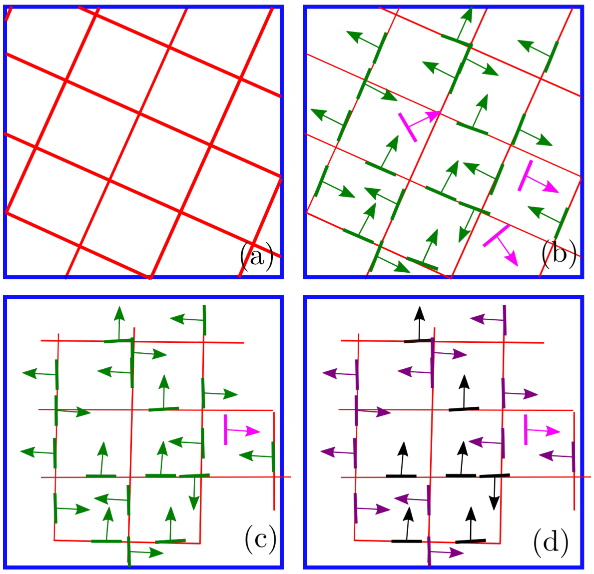
$$\hat{dd}_{horizontal} = \mathrm{L} \cdot \mathrm{median}\left(\frac{d_i}{L} - \mathrm{floor}\left(\frac{d_i}{L}\right)\right) \tag{4.6}$$

with

$$(d_i, \alpha_i) \in \text{horizontal lines}$$

The same is repeated for vertical lines and the parameters are then expressed $\mathbf{y} = (\hat{dd}_{horizontal}, \hat{dd}_{vertical}, \hat{\alpha})$. The described steps can be better visualized in image 4.3.

Figure 4.3: Steps of parameter extraction. (a) lines seen; (b) orientation computation (c) and (d) orientation filtering, correction and bags formation.



The idea of this method and the image were suggested by professor Jaulin.

5 TILE BASED LOCALIZATION

The localization is done in two main steps: the state prediction and the state contraction. The first one simply evolves the last estimation of the robot according to the state equations, explained in section 5.2, which increases the uncertainty; while the second one contracts that prediction respecting the constraints that come from the view and position equivalences, explained in subsection 3.2.2.

5.1 Contractor network

A contractor network is created using the equivalency relations explained in 3.2.2. The network is composed of constraints with respect to the observed tile parameters that must be respected by the the state. Both the state and the observation are described by three-dimensional interval vectors, that are inflated according to fixed errors in order to respect the incertitude of the movement and of the parameter extraction.

A limitation of the method is the maximum inflation, used to represent the incertitude, which is bounded by $L/2$, as more than that would allow for a two disjoint feasible solutions boxes, that not being treated and the method would fail. This can be circumvented with a sufficiently precise model of the system, that even though prone to the accumulation of errors such as any odometry, the contraction at most of the frames would suffice. That precision is not necessarily difficult to achieve, as the actual size of the tiles is not bounded, two perpendicular lines in the view being enough for the estimation of the parameters. Increasing the execution power of the system will also allow for a more imprecise model, as more frames will be processed and that maximum error only has to be respected at each frame.

5.2 State estimation

The first step in the localization loop is to evolve the previous estimation according to an approximation of the robot's movement. That is done with the Dubin's car model, expressed in equation 5.1, and having access to the $\mathbf{u}$ vector from the robot's proprioceptive sensors. It is important to note that that is one of the sources of incertitude of the method, as they represent a measure of the velocity and the change in heading, being an unreliable long-term estimation of the pose when used alone.

$$\begin{cases} \dot{x}_1 = u_1 \cos x_3 \\ \dot{x}_2 = u_1 \sin x_3 \\ \dot{x}_3 = u_2 \end{cases} \tag{5.1}$$

Using the Euler's integration chain, equation 5.2, the interval vector is updated and the incertitude inserted.

$$x_{n+1} = x_n + \dot{x}_n \, \mathrm{dt} \tag{5.2}$$

After the prediction step, the contraction removes all unfeasible solutions, respecting the network of constraints, as explained in section 3.1. This process is better depicted in the following algorithm.

Algorithm 1 Main loop of the state estimation

```
Result : Estimation of the current state of the robot.
while true do
    state ← integrate the state in order to evolve it
    parameters ← process incoming image and extract parameters
    state ← contract state checking equivalency between it and the parameters
    if state is not empty then
        send updated state
    else
        skip cycle
    end
end
```

6 IMPLEMENTATION

The project has been implemented both using the ROS framework [20] and an independent testing platform created from scratch. At a first moment, the architecture of the system was thought and evolved according to past material, then a ROS implementation was envisioned from the same material, leading to something closer to the workings of a real robot. However, problems with isolating parts of the system for testing lead to the creation of a simplified platform, with extended manipulation capacity and modular operation.

6.1 Architecture

6.1.1 C2

The robot node was thought to center most of the connections, making the interface between the environment and the multiple parts of the robot. The commands sent to the actual robot come from a controller node, which has a PD control for following a target point. The command node generates waypoints when in autonomous mode. The true pose, the waypoint and the estimated state are sent to a viewer node for better visualization of what is happening.

Figure 6.1 below represents what has been implemented of the project, reflecting what was mentioned at the beginning of the section.

6.1.2 ROS

Custom messages were added in order to represent an interval state $\mathbf{X} \in \mathbb{I}^3$ and observation $\mathbf{y} \in \mathbb{I}^3$. The same implementation as above can be seen in the node graph in figure 6.2. Here, the individual channels of data can be seen and the cycle used in the localization loop.

Figure 6.1: Architecture of the developed system.

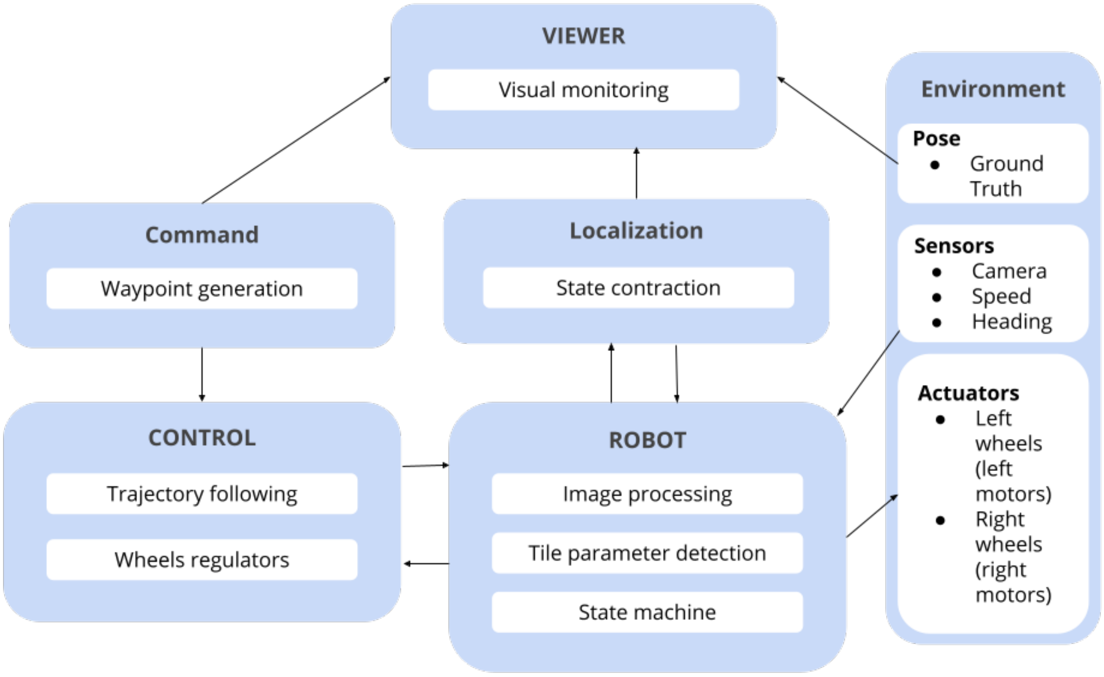
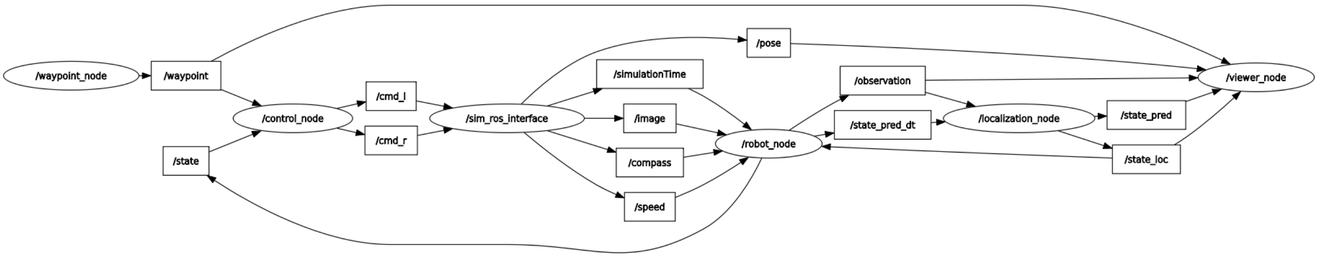


Figure 6.2: Node graph of the developed package.

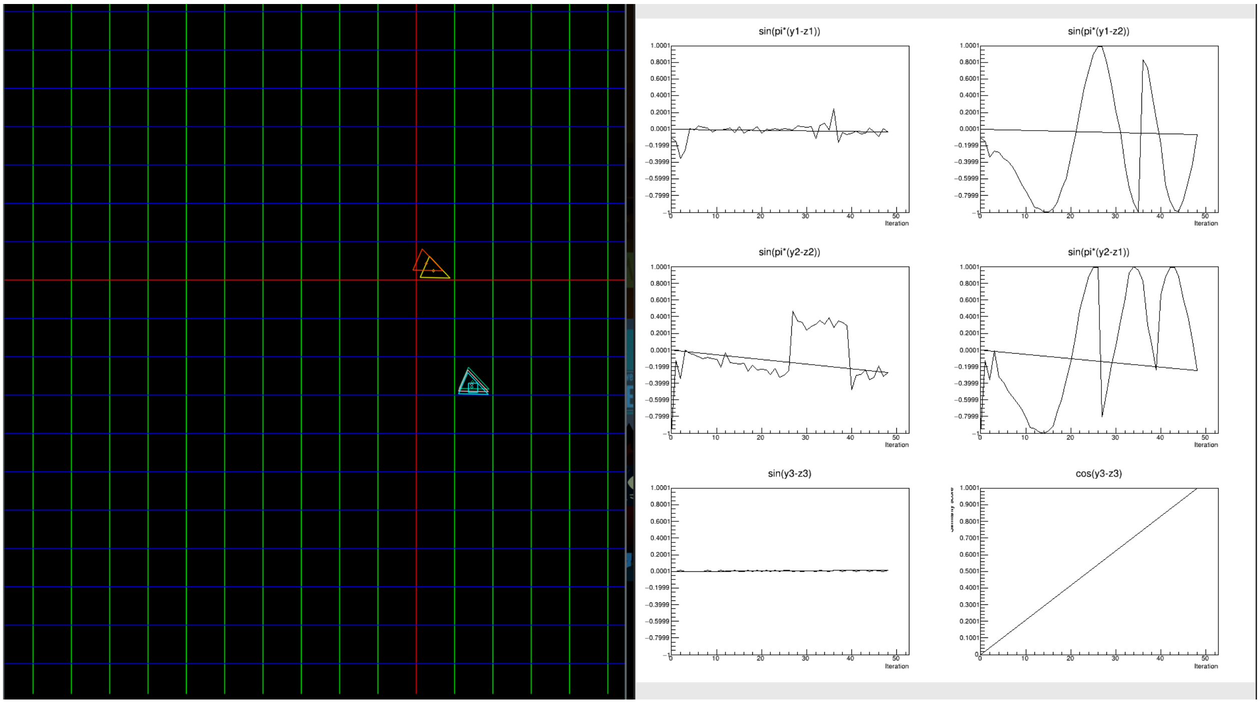


6.2 Testing environment

Because of the many moving parts in a ROS implementation, the need for a more controlled environment arose. A testing implementation was made removing all asynchronous communication and adding a custom made visualization of the global frame with the robot's pose, estimation, contraction and parameters in a single place. A view of some options is shown in figure 6.3, with different colors for each of the aforementioned positions, and the light blue rectangle is the contraction; it is noticeable that the pose, the prediction and the estimation are superposed around the contraction, in a successful localization. The graphs in the same image represent the equivalence equations discussed in 3.2.2, being desired that one of the columns stay around 0.

Modifiable options were added for it to be interactive, with visualization step by

Figure 6.3: Global frame, with center of the map shown in the red axis, and graph of the equivalence equations values.



step in both the global frame and the image processing, have a visualization in VIBES [21], and specify variables such as the number of pixels per meter, the size of the tiles and the output of a csv file with the equivalence values by frame. Seen in the following table 6.1.

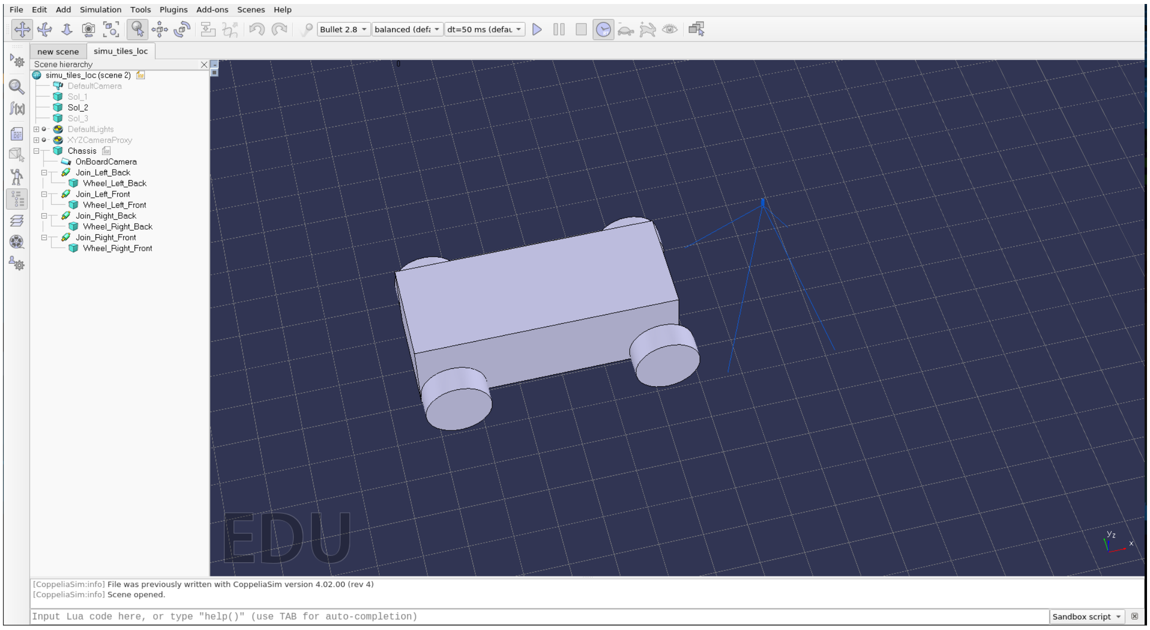
Table 6.1: Configurable parameters in the testing environment.

Parameter	Type	Description
interactive	Boolean	Enable frame by frame visualization
intervals	Boolean	Display intervals in VIBES
display	Boolean	Display steps of the image processing
ppm	Number	Set number of pixels per meter
tile-size	Number	Set size of the tiles in meters
output-file	String	Set testing data output

6.3 Simulation

A simulation of the described scenario was created from previous material made available, in a software called CoppeliaSim [22], correcting it and improving the communication with ROS, adapting it to better suit the project. Topics were added to provide the compass and the speed of the robot, being the vector **u** needed for the state prediction described in section 5.2. Those values could be estimated from an optical flow method or additional sensors, but as that is not the topic of the project, they were simply extracted from the simulated environment. The rectangle of the camera view can be seen in front of the robot. The simulated environment can be seen in figure 6.4.

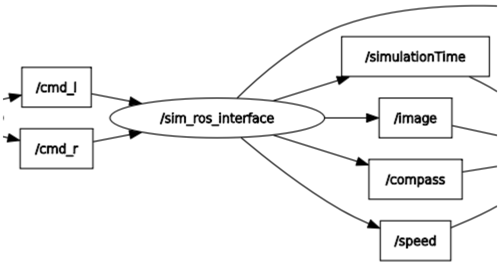
Figure 6.4: Simulation of the robot on a floor with tiles.



```
[CoppeliaSim:info] File was previously written with CoppeliaSim version 4.02.00 (rev 4)
[CoppeliaSim:info] Scene opened.
Input Lua code here, or type "help()" (use TAB for auto-completion)
```

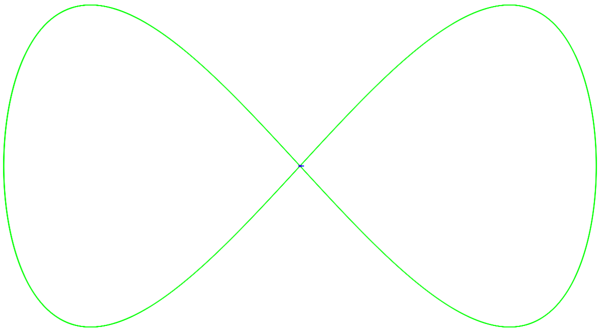
The robot uses differential steering, where the pair of wheels on each side had a different input. It moves according to its two inputs, with the PD control that was implemented relative to the error between the orientation of the robot and the line connecting the robot and the setpoint. The camera image, the simulation time for synchronizing communications, as well as the compass and speed for using in the state prediction are provided. The pose is available for debugging and visualization. The exchanges of data in the method are seen in figure 6.5.

Figure 6.5: Connections with the simulation in the proposed method.



An optional waypoint generation node was implemented, creating a setpoint for every time t that attracts the robot to follow a Lissajous curve, represented in figure 6.6. Choosing it or manual control depends on the launch file used.

Figure 6.6: Lissajous curve used as setpoint over time.



6.4 Docker

For the development and testing of the project in a controlled and isolated system, a Docker [23] image was created with the necessary software for dealing with ROS, image processing, interval analysis, among others, as well as a connection with the host machine for sharing code and data. This improves the reproducibility of the method and its results with an immutable and portable environment that works in any system supporting the Docker technology. It has been made available in https://github.com/birromer/ros-intervals-docker, with instructions for usage and modification.

7 RESULTS

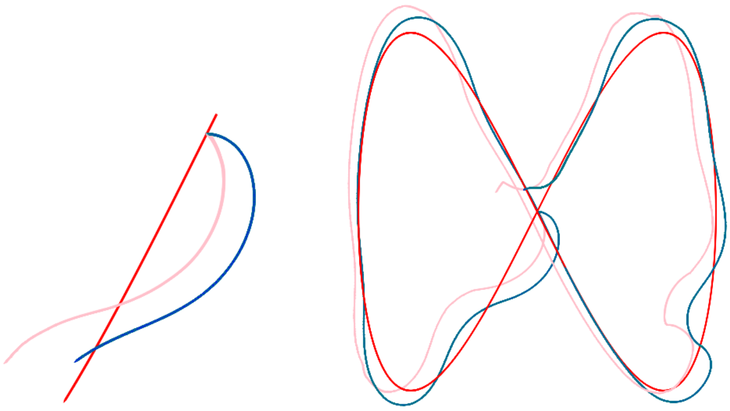
The current state of the method has achieved many interesting results, noticeably in validating the developed theory and a working localization in a limited scenario. Even though the localization has guaranteed properties because of the interval analysis background, it is still susceptible to failure due to incorrect parameters perceived in the image processing step. This vulnerability has been experienced in most long distance experiments and some resistance to it is yet to be implemented, such as a detection of outliers.

The evaluation has been performed in three different scenarios, first in the testing environment, described in section 6.2, where nothing but the correction of the method influenced the results, and later in a simulation, described in section 6.3, with the entire system working with ROS, having only the real robot replaced by a virtual one. The results of each are seen next.

For all experiments, the same pattern of tiles was used, either directly in the simulation or from a dataset created from the execution of arbitrary paths, having the state and image at every time step saved. The same starting position is used, which is considered known, as well as the same camera position and distance, also being assumed known its characteristics.

Figures 7.1 can be used as a baseline, displaying the robot being localized with nothing but the odometry, depending completely on the prediction step. Those are the errors nullified when contracting the state.

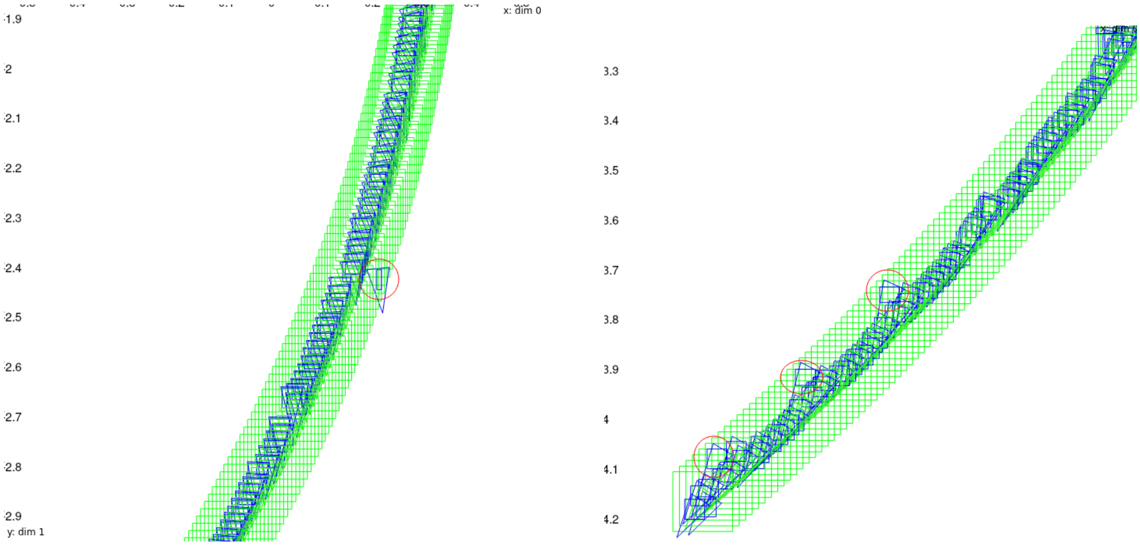
Figure 7.1: Examples of localization based only in odometry. The true pose is displayed in pink, the estimation in blue and the waypoints in red.



7.1 Testing environment, truth data

In this scenario, the objective was to validate the parameter extraction and the correct contraction of the state, without dealing with possible mistakes and carrying the resulting estimation of each cycle to the next. For this, at the beginning of each loop, the ground truth values were taken alongside each image in the dataset as the state and the maximum possible uncertainty was added by inflating the interval state vector, as explained in 5.1.

Figure 7.2: Working parameter extraction and state contraction in testing environment. Incorrect contractions circled red.



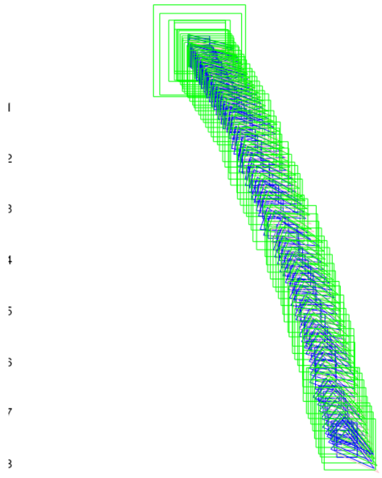
An example of performances is seen in figure 7.2. The green boxes are the initial uncertain area, created at each loop from the true pose, the smaller blue box is the state contraction made possible with equivalent tile parameters. Most of the contractions can be seen correct as they are at the center of the initial green area and considerably small, in the order of a few centimeters, corresponding to the confidence of the equivalence. A few examples of incorrect contractions are marked red, and they are a consequence of wrong parameters.

7.2 Testing environment, autonomous

In this case, the true position is only made available at the first iteration of the method, being all estimations carried and the error necessarily contained by successful contractions for it to work. At each time step a new image is provided, as well as a rough estimation of the speed and orientation, the same amount of information that would be available in the intended application.

In practice, the behaviour described before of occasional failures to extract parameters from the tiles view continues and leads to incorrect estimations of the state. However, while this event does not occur, the localization proceeds correctly, as seen in figure 7.3, where multiple cycles of prediction and contraction show themselves on top of the ground truth, distinguishable by the different levels of uncertainty at the prediction step in green, different from the boxes of constant size when using the truth data in the previous section. Unfortunately, this is only possible in short trajectories.

Figure 7.3: Working localization in testing platform in short trajectory.

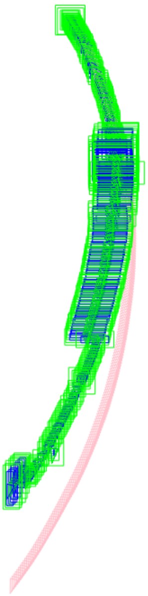


7.3 Simulation, truth data

This test case is done using ROS, which has 5 nodes with tasks running asynchronously and communication with an external simulator. Nothing but the initial position and the data from the sensors described in 6.3 is provided, which are extracted from the simulated robot with added noise. The complexity of the system reveals more of the maturity of the system, as no new apparent problems were introduced, being still limited by the consistency of the image processing.

Figure 7.4 shows the effects of a lost localization, where, after working in an initial slice of the path, incorrect parameters lead to failed contractions that make the robot use only the prediction, which is the implemented alternative for failed localization cycles, where the uncertainty is simply propagated. After some movement, it manages to contract again, but in the wrong position, due to the ambiguities in the tiled pattern, being not possible to recover at this point.

Figure 7.4: Method on simulator, lost after some meters. State predictions in green, state contractions in blue and ground truth in pink.



8 CONCLUSION AND FUTURE WORK

The development of new localization methods for mobile robots has considerable importance as it is an active research area with most of the problems still open. The usage of less traditional approaches, such as interval analysis, sheds interesting views to an area typically dominated by probabilistic methods and this application is useful in displaying some of its qualities, such as the guaranteed solutions and contractor operators that are capable of simplifying the size the challenge.

The presented method takes advantage of guarantees from a targeted scenario, more specifically the repetition and ambiguity of an industrial tiled floor, which is defined by non-linear relations that are elegantly and efficiently described as an intervalar method, leading to a method capable of localizing a robot by eliminating the growing incertitude over time. In the current state, the theoretical background of the method has been proved to work and its working is limited only by a faulty parameter detection, where everything proceed as expected when no errors on this part arise.

Beyond the image processing, the most pressing constraint to the localization is the quality of the state prediction, made from an estimation of the movement using proprioceptive sensors on the robot. Besides using other methods for estimating the movement, such as a visual odometry or alternative sensors, less radical options can be chosen, such as moving slower for better detecting parameters with some consensus. Using faster machines as well as increasing the size of the tiles and the field of view of the camera may also improve the results.

This project can be evolved with a more robust extraction of the displacement from images, including with some level of outlier detection and better fallback options for when the localization fails. A comparison with other methods, such as Monte Carlo Localization, is interesting too, as this would better place its performance within the area. Also, a thorough statistical evaluation should also be envisioned in order to better establish its fulfilment of objectives in localization missions.

REFERENCES

S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual Place Recognition: A Survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

Q. Brefort, L. Jaulin, M. Ceberio, and V. Kreinovich, "Towards Fast and Reliable Localization of an Underwater Object: An Interval Approach," p. 12.

S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents).* The MIT Press, 2005.

Hakyoung Chung, L. Ojeda, and J. Borenstein, "Sensor fusion for mobile robot dead-reckoning with a precision-calibrated fiber optic gyroscope," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 4, (Seoul, South Korea), pp. 3588–3593, IEEE, 2001.

R. D. Q. Maffei, "DP-SLAM," p. 95.

W. Zhen, S. Zeng, and S. Soberer, "Robust localization and localizability estimation with a rotating laser scanner," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, (Singapore, Singapore), pp. 6240–6245, IEEE, May 2017.

A. Heilig, I. Mamaev, B. Hein, and D. Malov, "Adaptive particle filter for localization problem in service robotics," *MATEC Web of Conferences*, vol. 161, p. 01004, 2018.

S. Rohou, B. Desrochers, and L. Jaulin, "Set-membership state estimation by solving data association," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, (Paris, France), pp. 4393–4399, IEEE, May 2020.

E. Stenborg, C. Toft, and L. Hammarstrand, "Long-Term Visual Localization Using Semantically Segmented Images," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, (Brisbane, QLD), pp. 6484–6490, IEEE, May 2018.

R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras," Oct. 2016.

K. A. Tsintotas, L. Bampis, and A. Gasteratos, "Probabilistic Appearance-Based Place Recognition Through Bag of Tracked Words," *IEEE Robotics and Automation Letters*, vol. 4, pp. 1737–1744, Apr. 2019.

R. Neuland, J. Nicola, R. Maffei, L. Jaulin, E. Prestes, and M. Kolberg, "Hybridization of Monte Carlo and set-membership methods for the global localization of underwater robots," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Chicago, IL, USA), pp. 199–204, IEEE, Sept. 2014.

R. Neuland, M. Mantelli, B. Hummes, L. Jaulin, R. Maffei, E. Prestes, and M. Kolberg, "Robust Hybrid Interval-Probabilistic Approach for the Kidnapped Robot Problem," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, Apr. 2021.

L. Jaulin, M. Kieffer, O. Didrit, and É. Walter, “Interval Analysis,” in *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics* (L. Jaulin, M. Kieffer, O. Didrit, and É. Walter, eds.), pp. 11–43, London: Springer, 2001.

P. Bao, L. Zhang, and X. Wu, “Canny edge detection enhancement by scale multiplication,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1485–1490, Sept. 2005.

“Image Transforms - Hough Transform.” https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm.

P. Mukhopadhyay and B. B. Chaudhuri, “A survey of Hough Transform,” *Pattern Recognition*, vol. 48, pp. 993–1010, Mar. 2015.

“ROS.org | Powering the world’s robots.” https://www.ros.org/.

V. Drevelle and J. Nicola, “VIBes: A Visualizer for Intervals and Boxes,” *Mathematics in Computer Science*, vol. 8, pp. 563–572, Sept. 2014.

“CoppeliaSim - Coppelia Robotics.” https://www.coppeliarobotics.com/coppeliaSim.

“Empowering App Development for Developers | Docker.” https://www.docker.com/.

RAPPORT D'EVALUATION
ASSESSMENT REPORT

Merci de retourner ce rapport par courrier ou par voie électronique en fin du stage à :
At the end of the internship, please return this report via mail or email to:

ENSTA Bretagne – *Bureau des stages - 2 rue François Verny - 29806 BREST cedex 9 – FRANCE*
☎ 00.33 (0) 2.98.34.87.70 / stages@ensta-bretagne.fr

I - ORGANISME / *HOST ORGANISATION*

NOM / *Name* ENSTA-Bretagne, Lab-STICC, Robex

Adresse / *Address* 2 rue F. Verny, Brest

Tél / *Phone (including country and area code)*

Nom du superviseur / *Name of internship supervisor*
Luc Jaulin

Fonction / *Function* Professeur

Adresse e-mail / *E-mail address* lucjaulin@gmail.com

Nom du stagiaire accueilli / *Name of intern* Bernardo Hummes Flores

II - EVALUATION / *ASSESSMENT*

Veuillez attribuer une note, en encerclant la lettre appropriée, pour chacune des caractéristiques suivantes. Cette note devra se situer entre **A (très bien)** et **F (très faible)**
Please attribute a mark from ***A (excellent)*** *to* ***F (very weak).***

MISSION / *TASK*

- La mission de départ a-t-elle été remplie ? *Was the initial contract carried out to your satisfaction?* A [✓ B] C D E F

- Manquait-il au stagiaire des connaissances ? *Was the intern lacking skills?* ☐ oui/*yes* ☑ non/*no*

  Si oui, lesquelles ? / *If so, which skills?*

ESPRIT D'EQUIPE / *TEAM SPIRIT*

- Le stagiaire s'est-il bien intégré dans l'organisme d'accueil (disponible, sérieux, s'est adapté au travail en groupe) / *Did the intern easily integrate the host organisation? (flexible, conscientious, adapted to team work)* [✓ A] B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here*

COMPORTEMENT AU TRAVAIL / *BEHAVIOUR TOWARDS WORK*

Le comportement du stagiaire était-il conforme à vos attentes (Ponctuel, ordonné, respectueux, soucieux de participer et d'acquérir de nouvelles connaissances) ?
Did the intern live up to expectations? (Punctual, methodical, responsive to management instructions, attentive to quality, concerned with acquiring new skills)?

[✓ A] B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* ______

INITIATIVE – AUTONOMIE / *INITIATIVE – AUTONOMY*

Le stagiaire s'est –il rapidement adapté à de nouvelles situations ? [✓ A] B C D E F
(Proposition de solutions aux problèmes rencontrés, autonomie dans le travail, etc.)

Did the intern adapt well to new situations? [✓ A] B C D E F
(eg. suggested solutions to problems encountered, demonstrated autonomy in his/her job, etc.)

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* ______

CULTUREL – COMMUNICATION / *CULTURAL – COMMUNICATION*

Le stagiaire était-il ouvert, d'une manière générale, à la communication ? [✓ A] B C D E F
Was the intern open to listening and expressing himself /herself?

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* ______

OPINION GLOBALE / *OVERALL ASSESSMENT*

- La valeur technique du stagiaire était : [✓ A] B C D E F
  Please evaluate the technical skills of the intern:

III - PARTENARIAT FUTUR / *FUTURE PARTNERSHIP*

- Etes-vous prêt à accueillir un autre stagiaire l'an prochain ?
  Would you be willing to host another intern next year? ☑ oui/*yes* ☐ non/*no*

Fait à Brest, le 29 aout 2021
In ______, on ______

Signature Entreprise Luc Jaulin Signature stagiaire
Company stamp ______ *Intern's signature*

Merci pour votre coopération
We thank you very much for your cooperation